CLAIMS

What is claimed is:

1. A computer-based event handling system, comprising:

a framework component that supplies classes of objects that can raise events; and

a synchronization component that controls in part synchronization of access to data based on categorization of at least one of objects and instances defined by the framework.

2. The system of claim 1, the framework is supplied by an operating system or as a library for use by the operating system, wherein at least one of the framework and synchronization component automatically manages or serializes the events in order that a client component can process client-specific tasks.

3. The system of claim 2, the client component is a device driver.

4. The system of claim 1, the events are managed by the framework or synchronization component to allow one or more aspects of the events to occur in a one-time manner and in accordance with a serialized process.

5. The system of claim 4, the events are processed as part of a request.

6. The system of claim 2, the object provides a handle to enable the client component to manipulate the object and request additional local memory to be allocated for processing client tasks.

7. The system of claim 6, further comprising a configuration component to enable the client component to disable or enable automated serialization and synchronization.

8. The system of claim 7, the configuration component includes one or more Application Programming Interfaces to facilitate selection of serialization and synchronization features.

9. The system of claim 1, the framework component provides events to a device driver though a series of callback functions registered by the device driver.

10. The system of claim 9, the device driver generally queues a handler or starts Input/Output (I/O) on the handler, marking state, and then returns.

11. The system of claim 1, the framework component is structured in accordance with state full objects that allow a device driver to register events, and provide API's.

12. The system of claim 1, the events are associated with a pipeline.

13. The system of claim 12, the pipeline includes at least one of a dispatch component, a plug and play component, a device model component, and an Input/Output component.

14. The system of claim 12, the pipeline employs a series of stages that a request traverses in the processing of a Driver Model request or an I/O Request Packet (IRP).

15. The system of claim 14, further comprising objects within each stage of the pipeline that raise an event to a device driver through an event callback to allow the driver to have control of which action to take, or provide some default action that may result in the requests completion, or forwarding to the next stage.

16. The system of claim 15, further comprising a processing component to determine whether a request reaches an end of the pipeline without being accepted by the device driver or by automatic processing in a respective stage, otherwise a default processing of the request occurs.

17. The system of claim 16, the default processing depends on whether the driver is configured as a Filter Driver in which the request is forwarded to a next lower device driver, or a non-filter driver in which the request is completed with a STATUS_INVALID_DEVICE_REQUEST.

18. The system of claim 1, further comprising one or more data packets that are employed to facilitate communications between the components *via* at least one of a local processing system, a local network system, and a remote network system.

19. A computer readable medium having computer readable instructions stored thereon for implementing the framework component and the synchronization component of claim 1.

20. A computer-based event processing system, comprising:
means for interacting with an object associated with a device model framework;
means for generating at least one event within the framework;
means for processing the event; and
means for automatically serializing the event in accordance with the framework.

21. The system of claim 20, further comprising means for configuring automatic serialization of the event.

22. A method to facilitate event processing in accordance with an operating system, comprising:

defining at least one rule for processing an event;

defining at least one object responsive to the event, the object categorized according to a framework; and

automatically serializing the event in accordance with at least one other event.

23. The method of claim 22, further comprising automatically serializing the event in accordance with at least one of an operating system process and the framework.

24. The method of claim 22, further comprising enabling a client component to disable automatic serialization of events.

25. The method of claim 24, further comprising providing a local memory to facilitate client-specific processing.

26. A computer readable medium having a data structure stored thereon, comprising:

a first data field related to a framework object having a handle associated therewith to facilitate data access;

a second data field that relates to a client component that communicates *via* the handle and receives events from the framework object; and

a third data field that serializes the events.

27. The computer readable medium of claim 26, further comprising at least one Application Programming Interface (API).

28. The computer readable medium of claim 27, the API includes at least one of a scope object, a scope device and a scope specified data structure.

29 The computer readable medium of claim 27, the API includes at least one of an execution level passive, an execution level dispatch, and an execution level specified data structure.

30. The computer readable medium of claim 27, further comprising an acquire lock data structure and a release lock data structure.